IMPLEMENTING THE CC-USB CONTROL MODULE FOR USE IN CAMAC CRATES AT THE FERMILAB TEST BEAM FACILITY

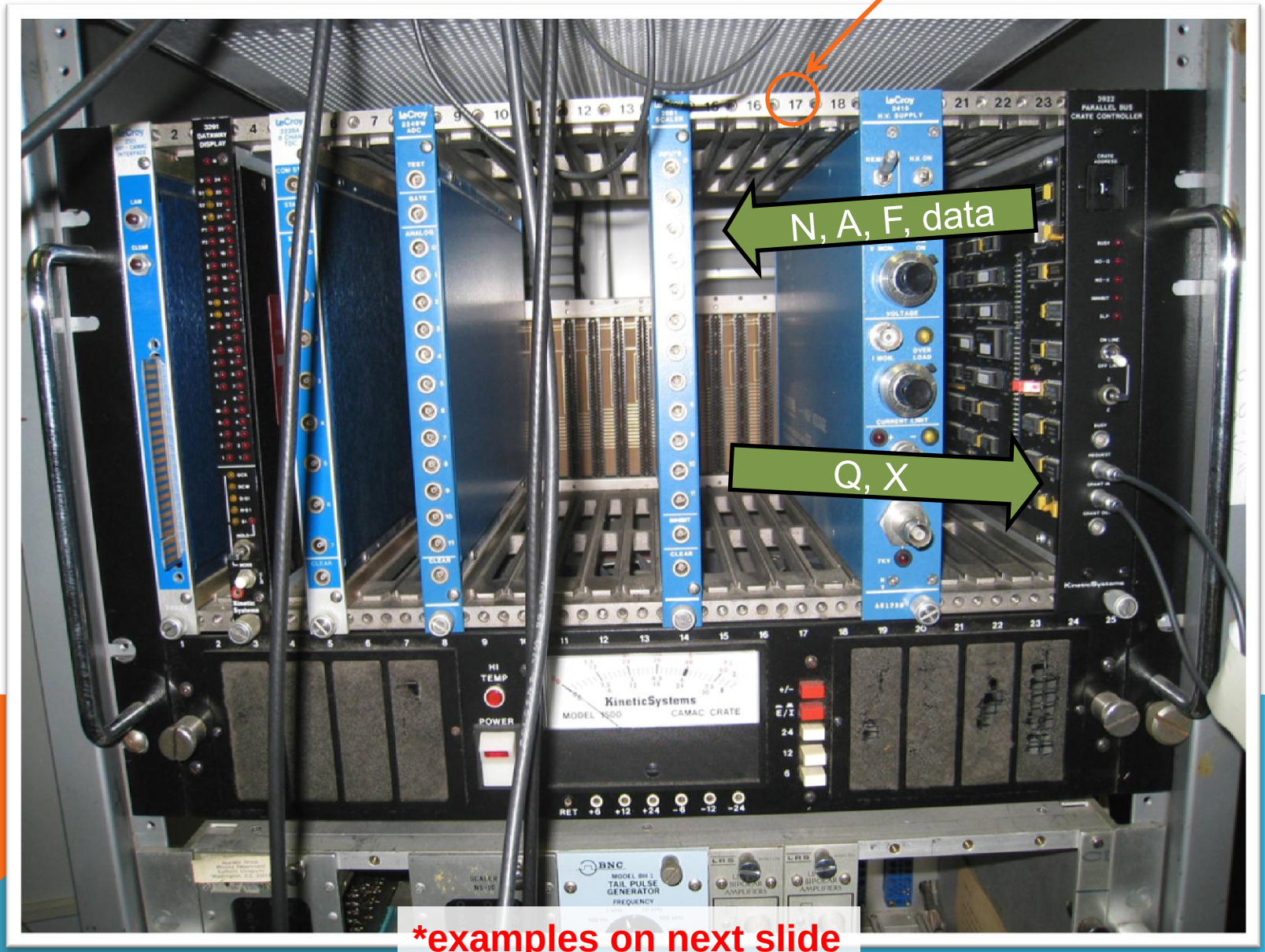
BY KAREN LIPA, SIST INTERN

AUG 8, 2012

OUTLINE

- Background info
- Specific use: cosmic ray test stand
- Goals
- My Process
- Questions

HARDWARE: CAMAC SYSTEM



N, A, AND F

N	A	F	Function	Data
0	*	16	Write a 16-bit marker word into the output data stream	16
1…24	*	*	Executes N(1..24) A(*) F(*) command on CAMAC data way	16/24
25	0	0	Read Firmware ID	32
25	1	0	Read Global Mode	16
25	1	16	Write Global Mode	16
25	2	0	Read Delays	16
25	2	16	Set Delays	16
25	3	0	Read Scaler Readout Control	24
25	3	16	Write Scaler Readout Control	24
25	4	0	Read User LED Source Selector	32
25	4	16	Write User LED Source Selector	32
25	5	0	Read User NIM Output Source Selector	32
25	5	16	Write User NIM Output Source Selector	32
25	6	0	Read Source Selector for User Devices	32
25	6	16	Write Source Selector for User Devices	32
25	7	0	Read Timing for Delay & Gate Generator A	32
25	7	16	Write Timing for Delay & Gate Generator A	32
25	8	0	Read Timing for Delay & Gate Generator B	32
25	8	16	Write Timing for Delay & Gate Generator B	32
25	9	0	Read LAM Mask	24
25	9	16	Write LAM Mask	32
25	10	0	Read CAMAC LAM (pseudo-register)	24
25	11	0	Read Scaler A (pseudo-register)	32
25	12	0	Read Scaler B (pseudo-register)	32
25	13	0	Read Extended Delays Register	32
25	13	16	Write Extended Delays Register	32
25	14	0	Read USB Buffering Setup Register	32
25	14	16	Write USB Buffering Setup Register	32
25	15	0	Read Broadcast Map (notepad register)	24
26	*	*	execute Broadcast A(*) F(*) on CAMAC dataway	16/24
27	**	**	Set Broad cast mask (3 sequential calls for 24 bit mask)	24
28	8	29	CAMAC Z	-
28	9	29	CAMAC C	-
29	9	24	Set CAMAC I	-
29	9	26	Clear CAMAC I	-

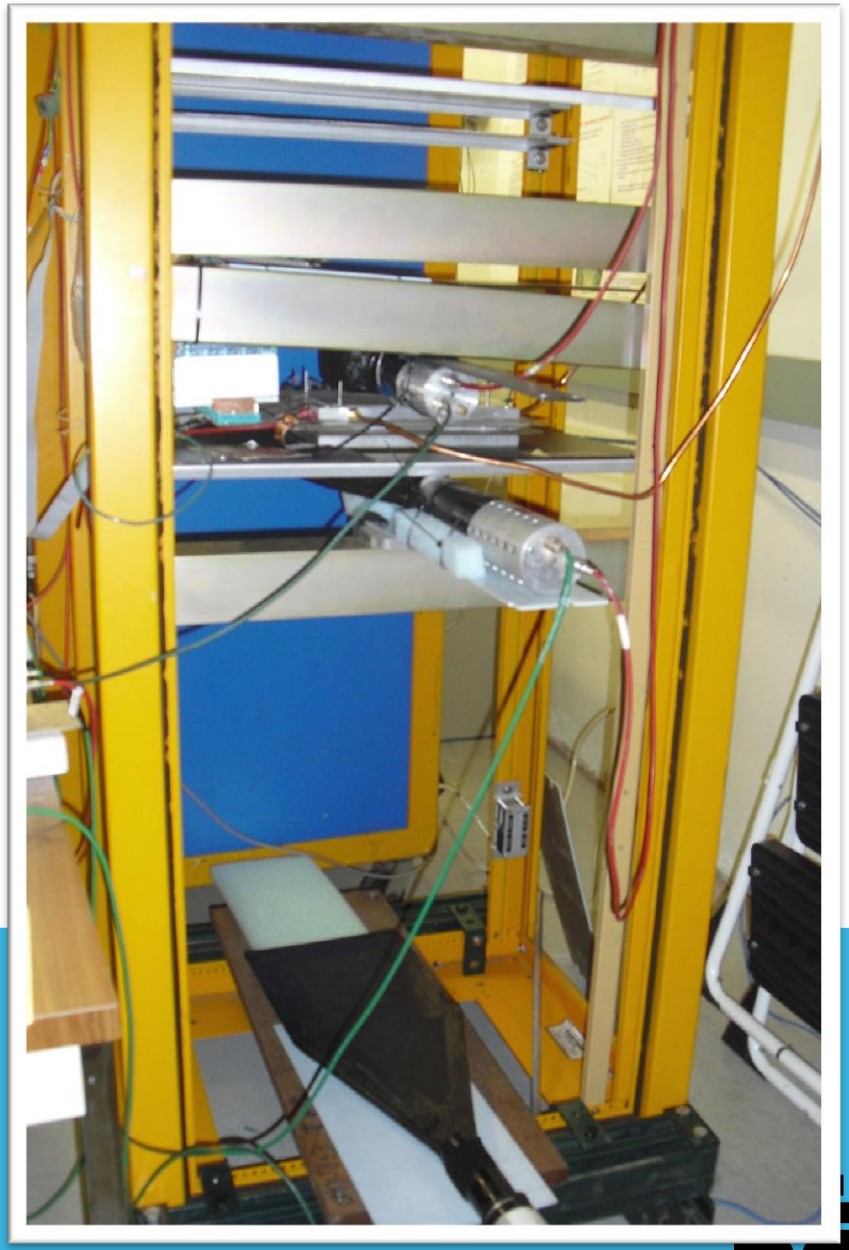
- Read (F 0-7)
- Write (F 16-23)
- I: inhibit
- Z: initialize
- C: clear

HARDWARE: CC-USB

- Control module
- Transmits and receives data to/from crate modules
- Accessed by computer via USB cable
- Newest type of control module (others are obsolete)
- Not previously equipped for use at Fermilab

COSMIC RAY TEST STAND PROJECT

- Goal: to produce a reliable means of testing new detectors
- Wire chambers – x- and y-plane wires indicate location of particles
- Scintillators and PMT's omit signal when particle travels through (trigger)



HARDWARE

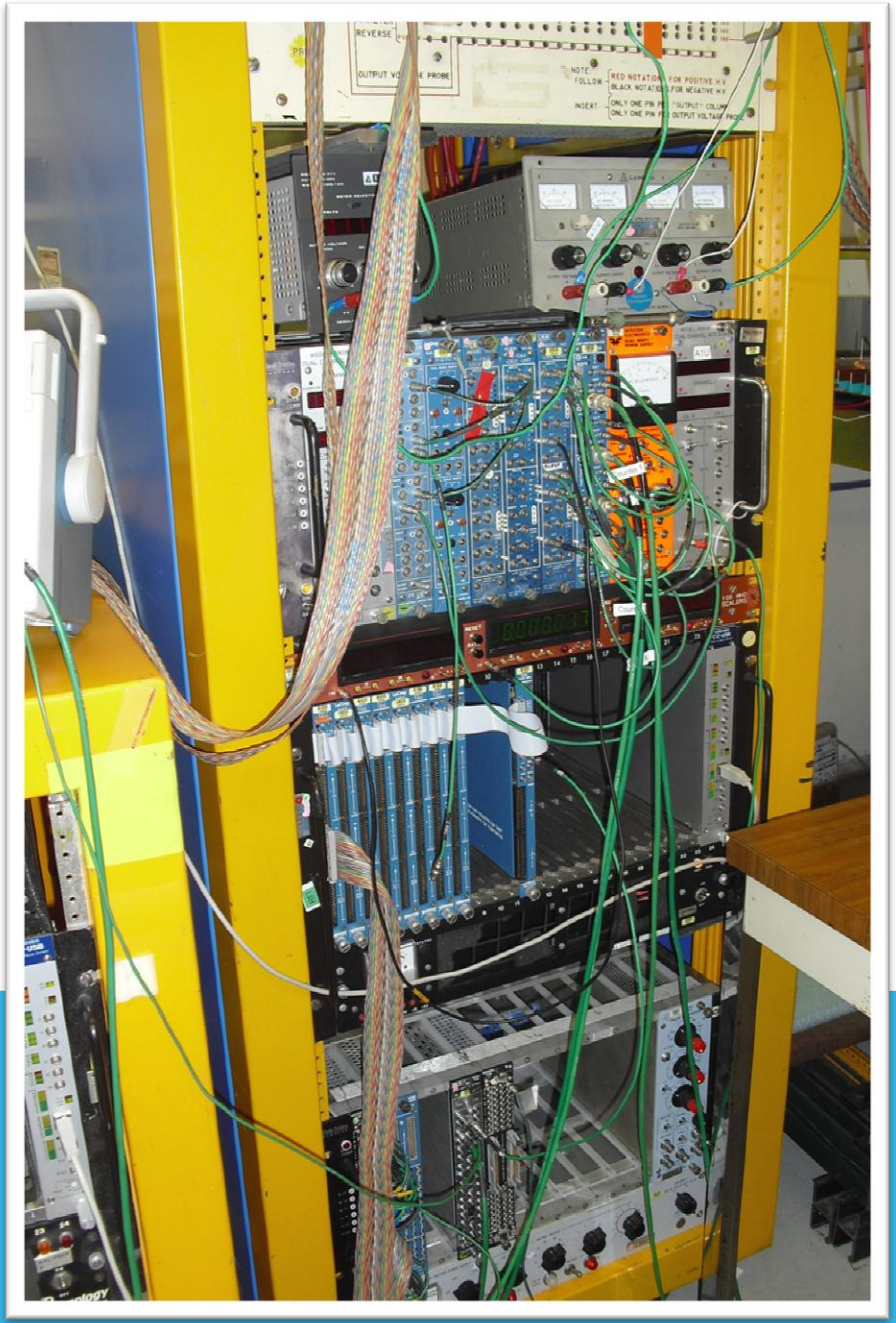
CAMAC crates

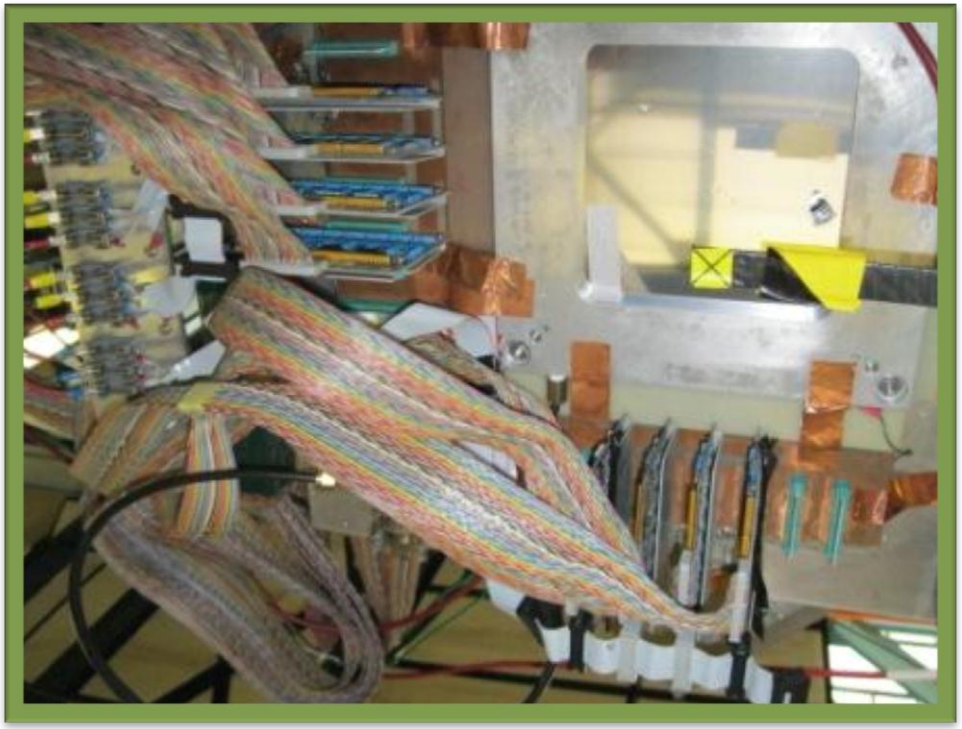
- Lots lying around, economical

CC-USB

- CAMAC parallel bus is now obsolete

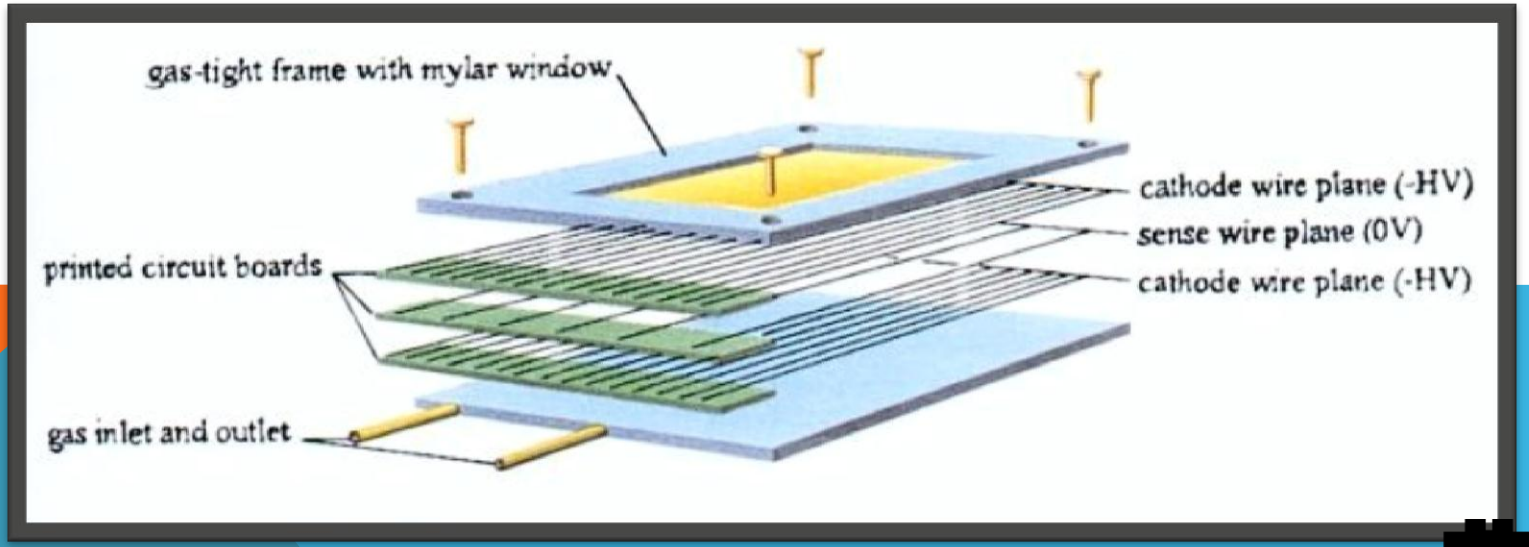
Wire chambers, scintillators, PMTs





HARDWARE: WIRE CHAMBERS

- x- and y-plane wires indicate locations where particles hit
- Sends a signal to card



Photos:
http://www-ppd.fnal.gov/FTBF/Facility/Inst_mwpc/Instructions.html
http://universe-review.ca/R15-20-accelerators.htm

HARDWARE: SCINTILLATORS + PMT'S

- scin·til·la·tor (s n tl- t r). n. A substance that glows when hit by high-energy particles or photons (www.freedictionary.com)

- Photomultiplier tubes: "multiply" result of scintillator hit (emit electrical signals)

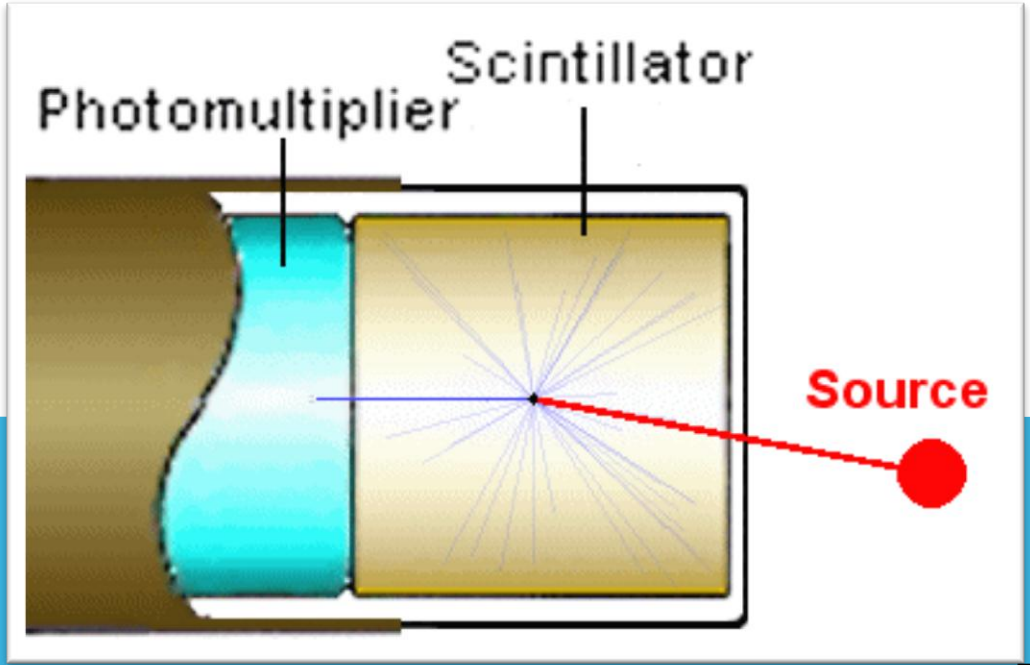
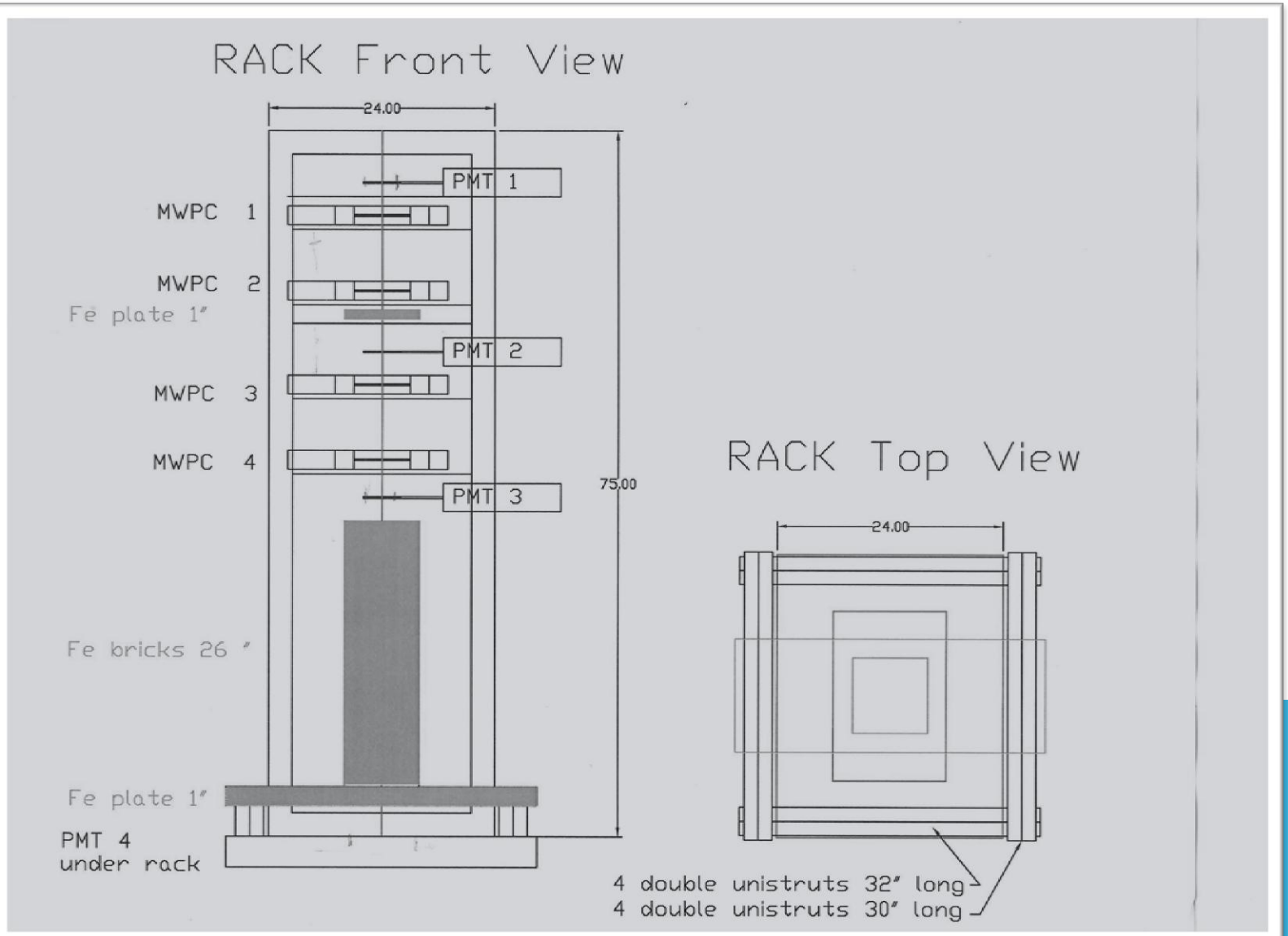


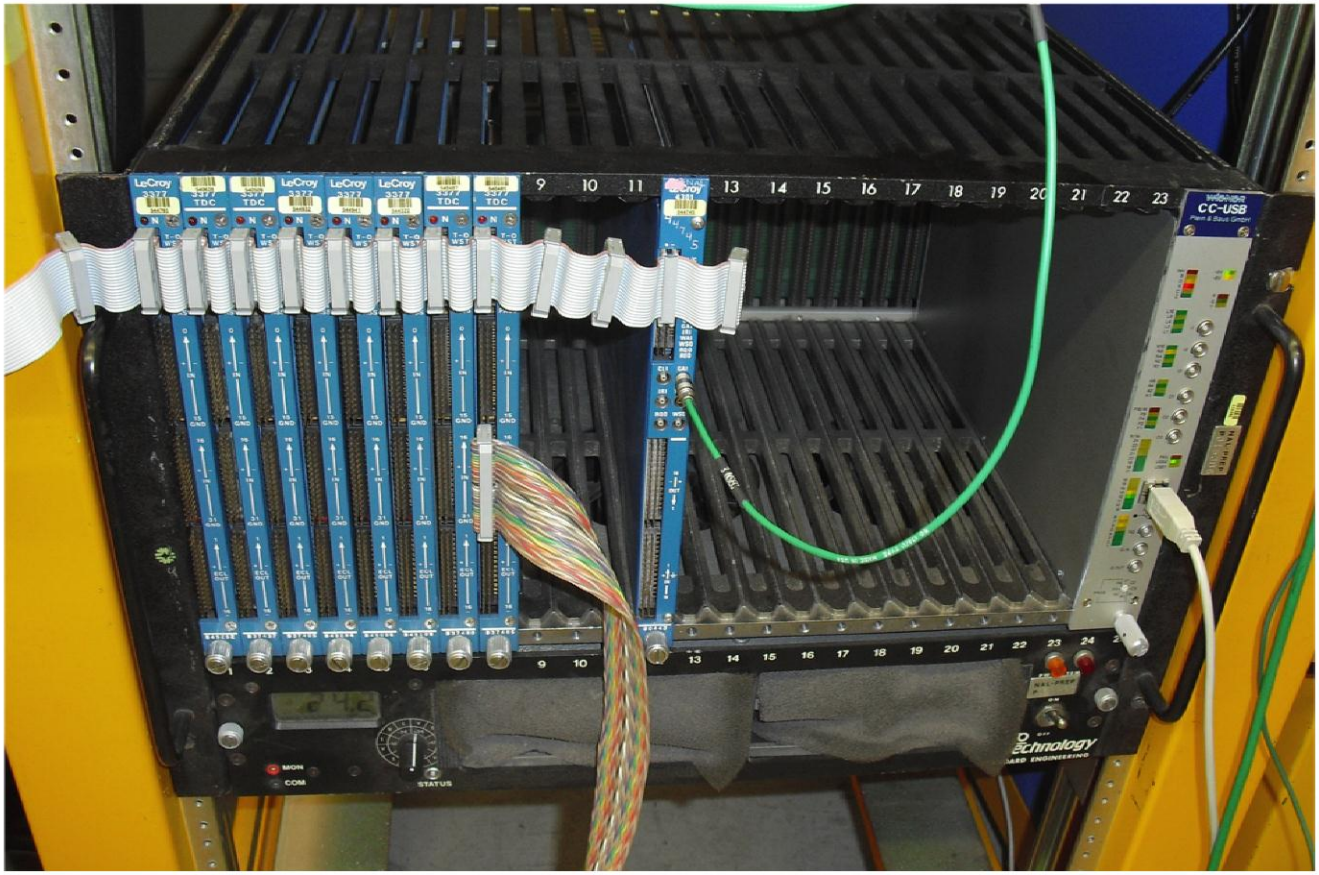
Photo: http://williamson-labs.com/ltoc/cbr-tech.htm

THE SETUP


RACK Front View
24.00
PMT 1
MWPC 1
MWPC 2
Fe plate 1"
PMT 2
MWPC 3
MWPC 4
PMT 3
75.00
Fe bricks 26 "
Fe plate 1"
PMT 4
under rack
RACK Top View
24.00
4 double unistruts 32" long
4 double unistruts 30" long

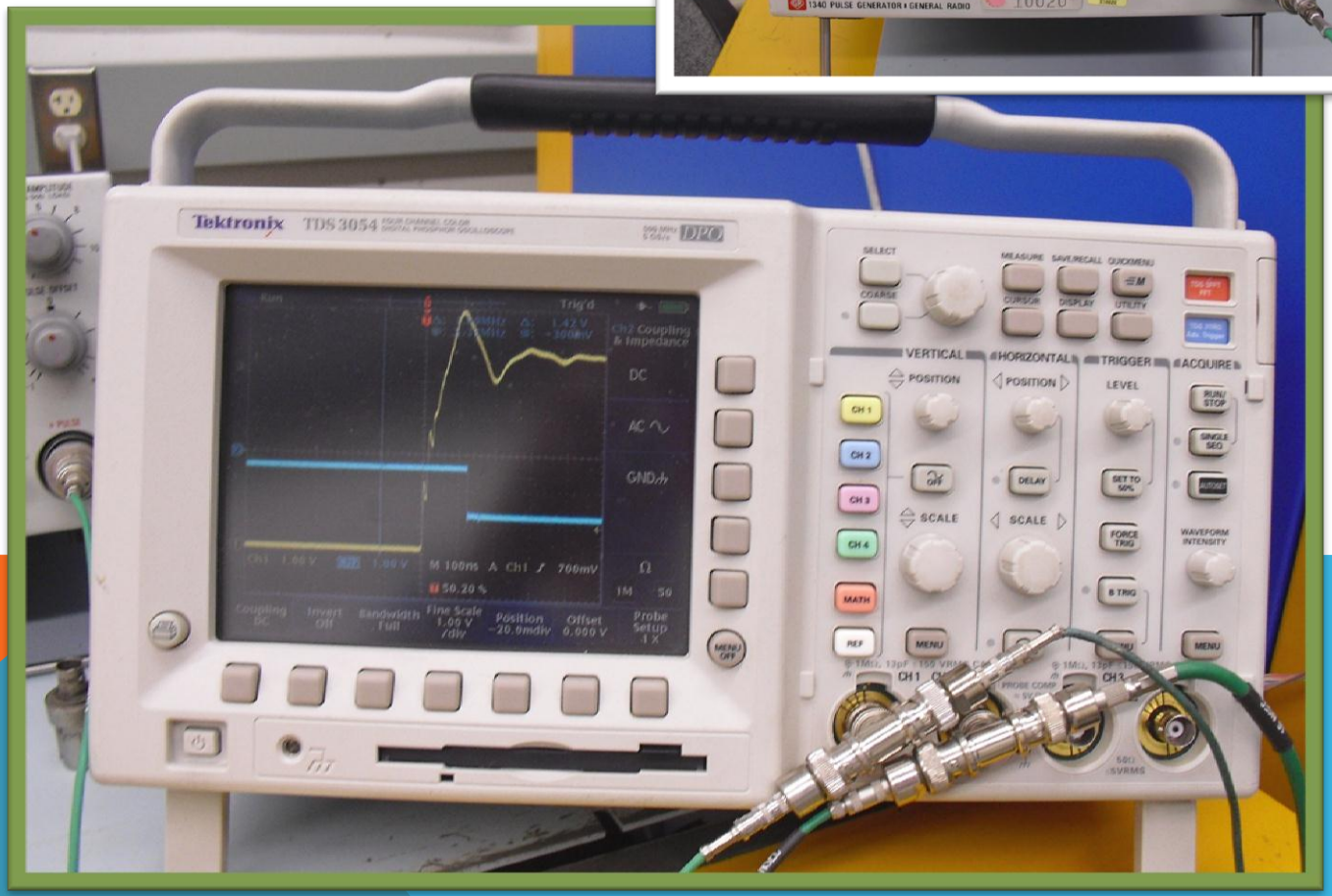
THE SETUP: CAMAC CRATE

- Lecroy 3377 Time-to-Digital converter
  - counts the time from a hit in the wire chamber until stop signal
- Lecroy 4301: encoding and readout
  - sends out stop signal to all the TDCs



TEST PROCEDURE

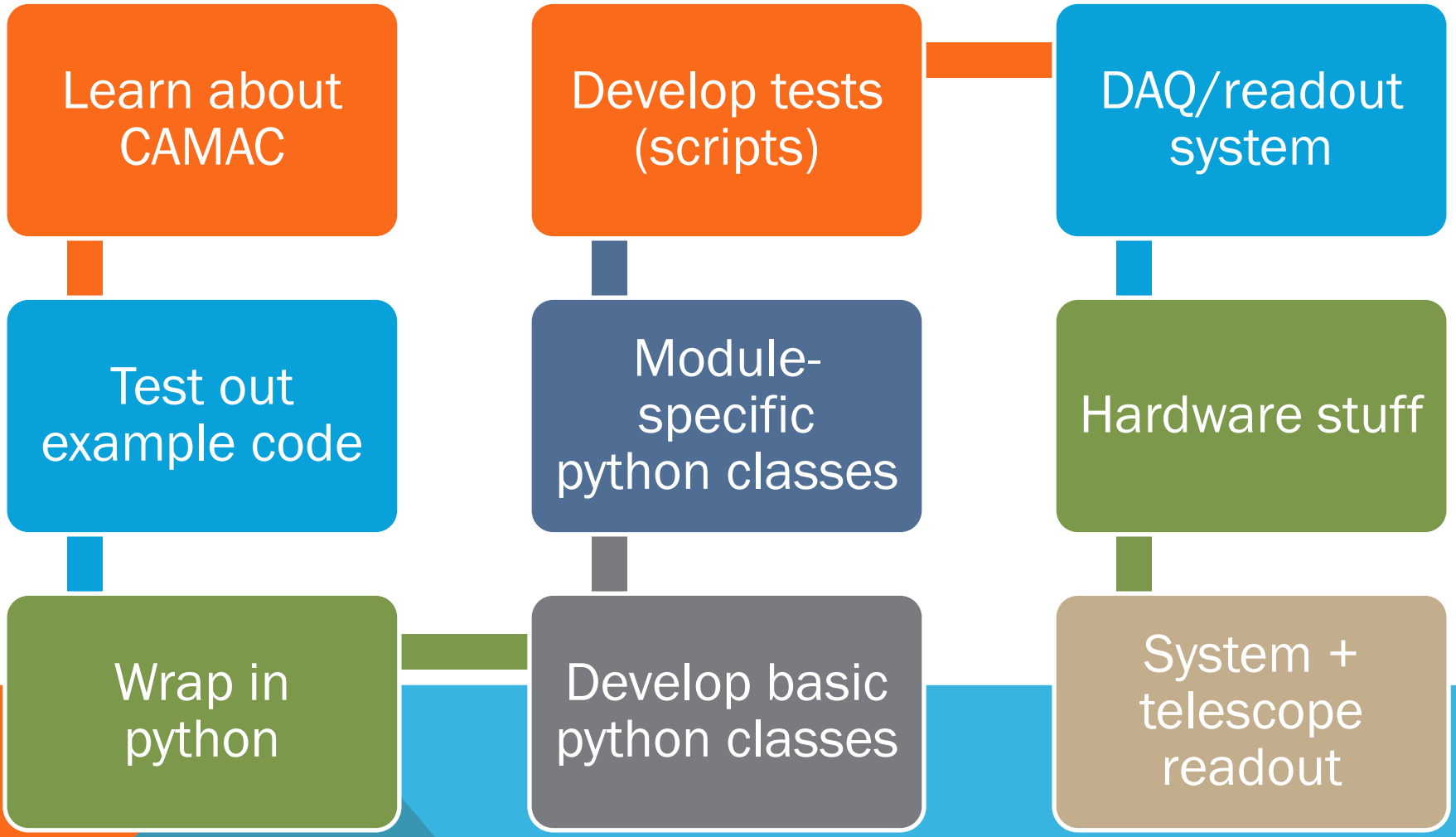
Pulse generator: produces mock STOP and hit signals



GOALS

- Software to support the use of the CC-USB control module
- Python wrapping (ease for user)
- Specifics relevant to cosmic ray project

STEPS I TOOK


Learn about CAMAC
Develop tests (scripts)
DAQ/readout system
Test out example code
Module-specific python classes
Hardware stuff
Wrap in python
Develop basic python classes
System + telescope readout

PYTHON WRAPPING

All the benefits of Python …

- Ease of use (syntax)**
- High level
- Object-oriented
- Script-based (testing)

With the power of C!

- Access to USB functions (examples)
- speed

HOW TO WRAP A C FUNCTION

STEP 1: WRAPPER FUNCTION

- Functions (PyArg_ParseTuple and Py_BuildValue) take in and return Python objects
- Within the wrapper function, call is made to C function defined elsewhere

```
static PyObject *
spam_system(PyObject *self, PyObject *args)
{
    const char *command;
    int sts;

    if (!PyArg_ParseTuple(args, "s", &command))
        return NULL;
    sts = system(command);
    return Py_BuildValue("i", sts);
}
```

Photo: http://docs.python.org/extending/extending.html#a-simple-example

HOW TO WRAP A C FUNCTION

STEP 2: METHODS TABLE

- Defines the “nickname” that each function will be called by

```
static PyMethodDef SpamMethods[] = {
    ...
    {"system",  spam_system, METH_VARARGS,
     "Execute a shell command."},
    ...
    {NULL, NULL, 0, NULL}        /* Sentinel */
};
```

Photo: http://docs.python.org/extending/extending.html#a-simple-example

HOW TO WRAP A C FUNCTION

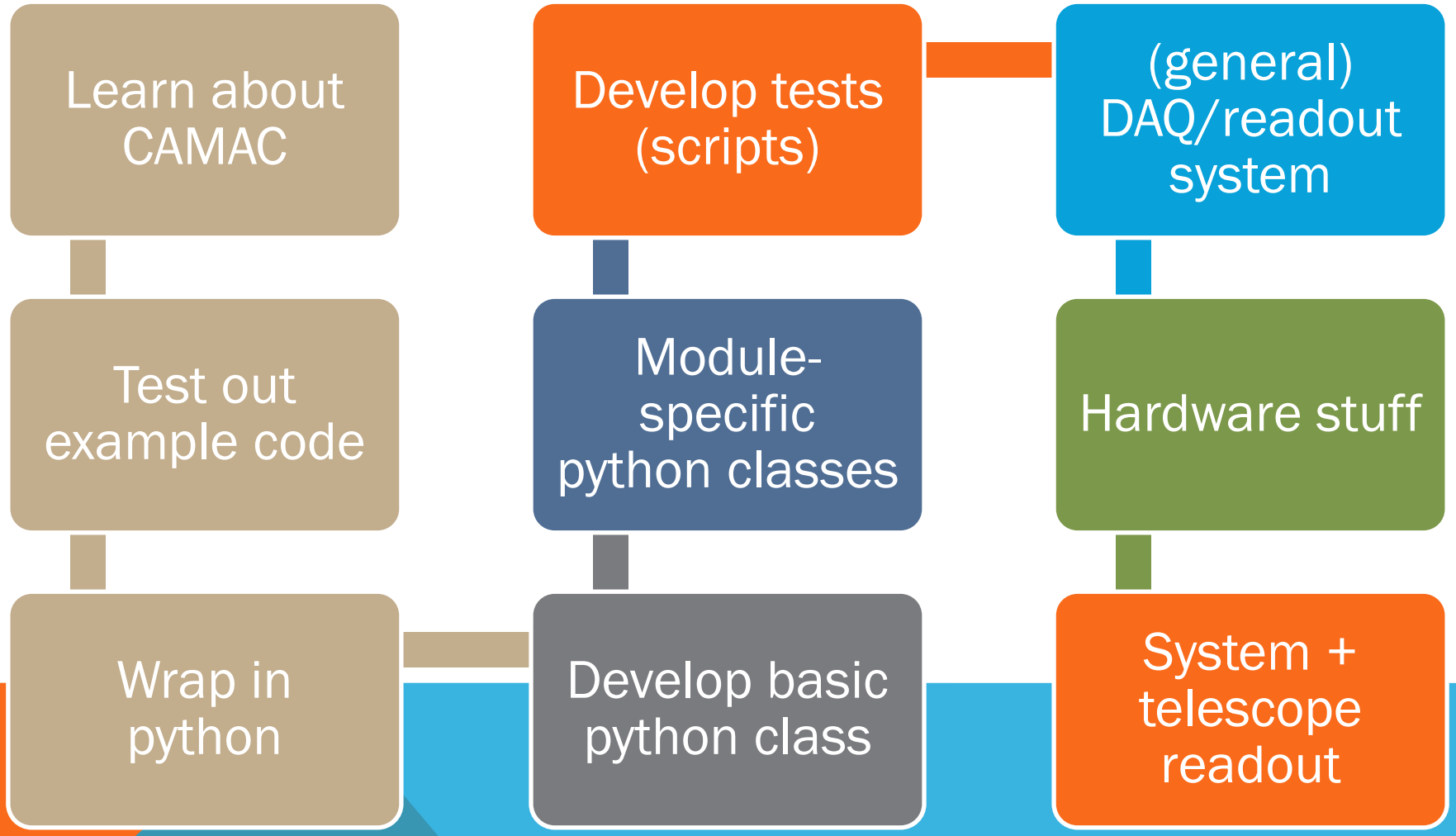
STEP 3: INITIALIZATION FUNCTION

- Defines the name of the module to be called from python

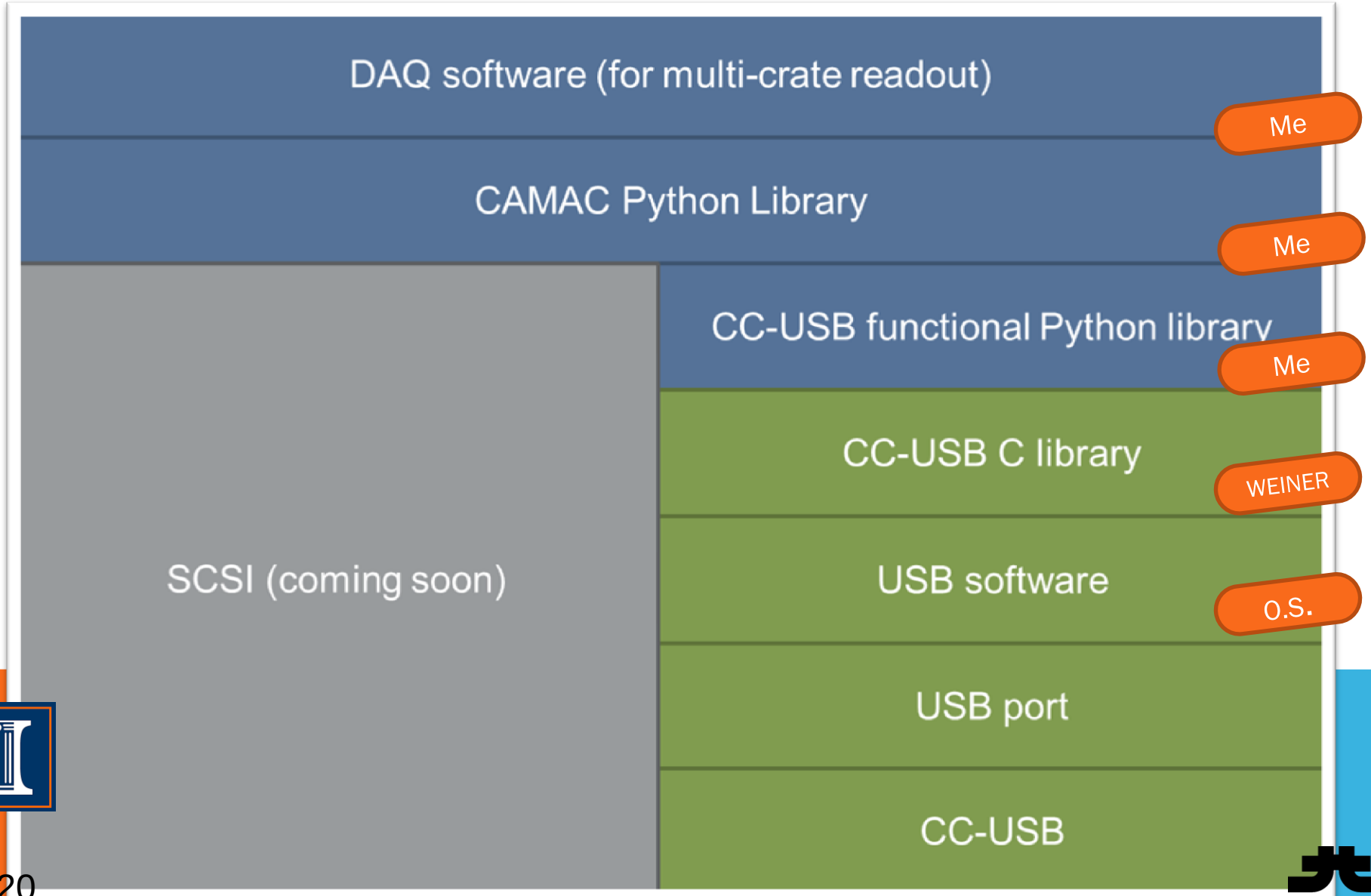
```
PyMODINIT_FUNC
initspam(void)
{
    (void) Py_InitModule("spam", SpamMethods);
}
```

Photo: http://docs.python.org/extending/extending.html#a-simple-example

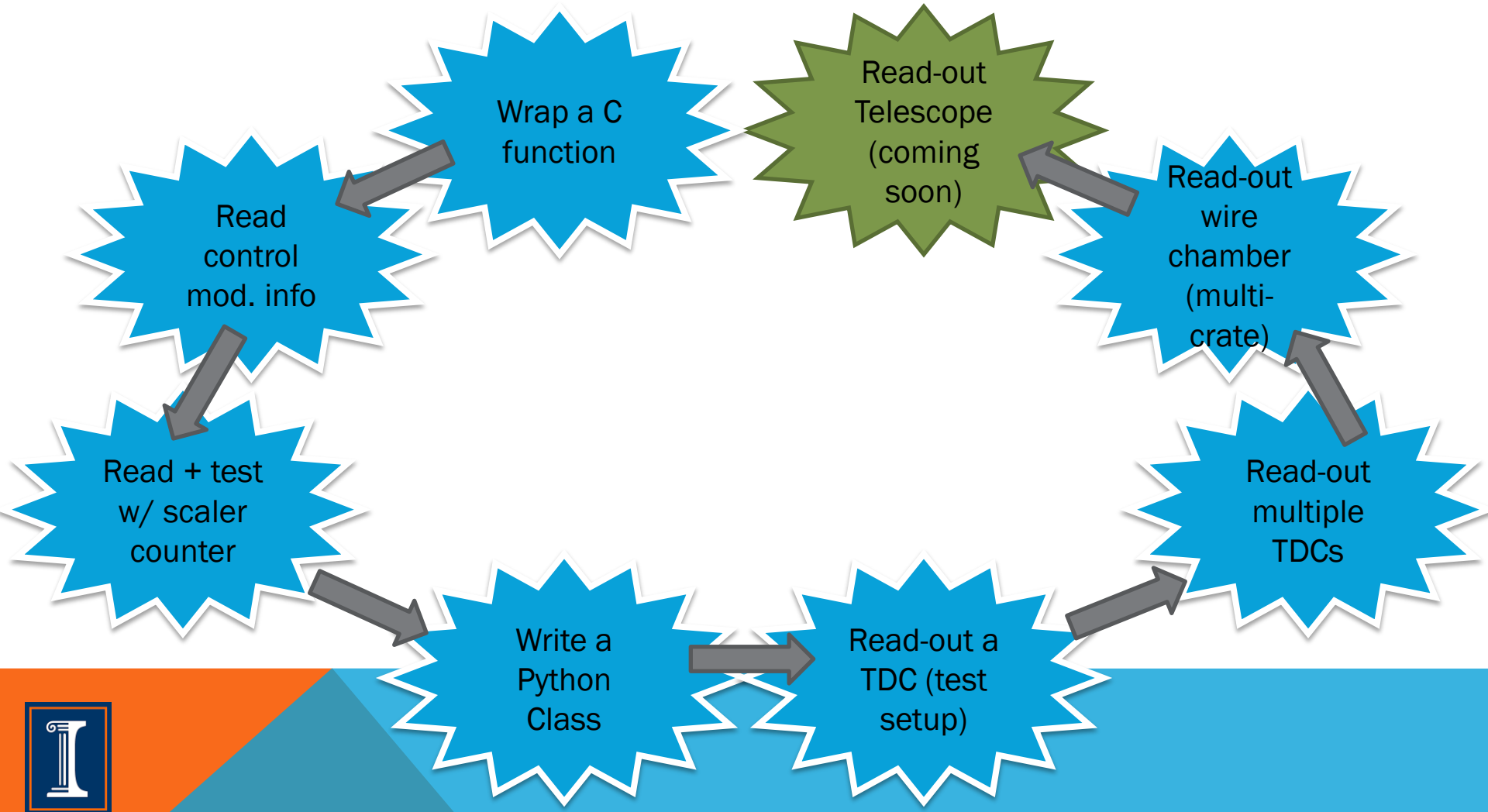
STEPS I TOOK


Learn about CAMAC
Develop tests (scripts)
(general) DAQ/readout system
Test out example code
Module-specific python classes
Hardware stuff
Wrap in python
Develop basic python class
System + telescope readout

MY CONTRIBUTION: SOFTWARE



BIG MILESTONES



ACKNOWLEDGEMENTS

- Supervisor: Geoff Savage
- Mentors: Elliott McCrory, Jamieson Olsen
- Dianne Engram, SIST committee
- Dr. James Davenport

QUESTIONS?